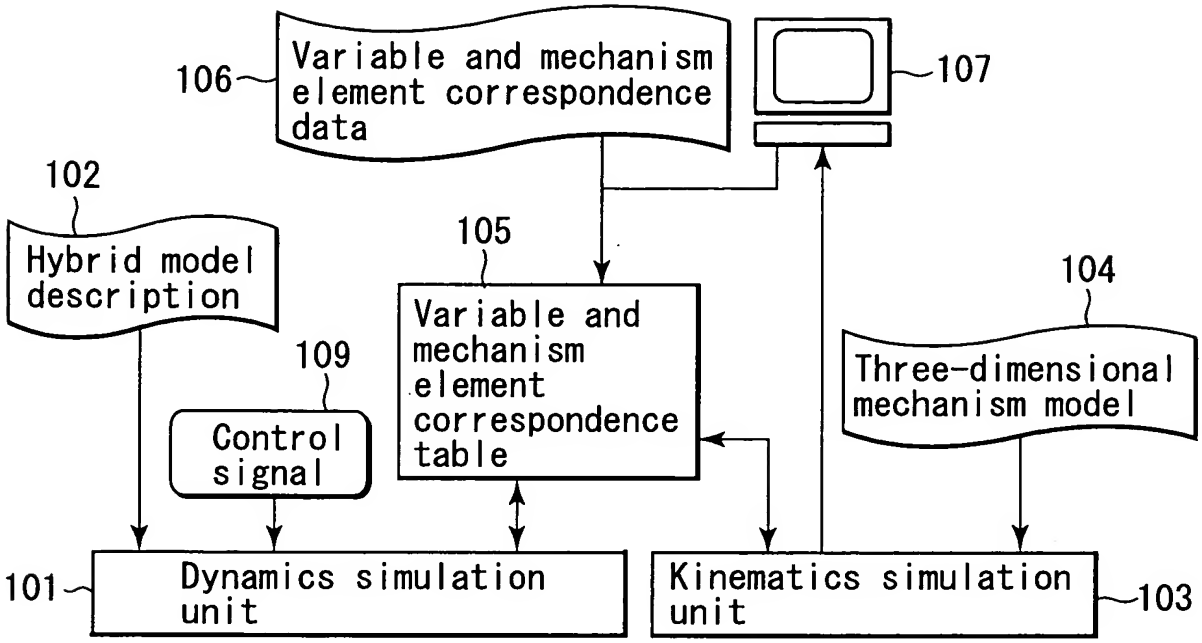


FIG. 1

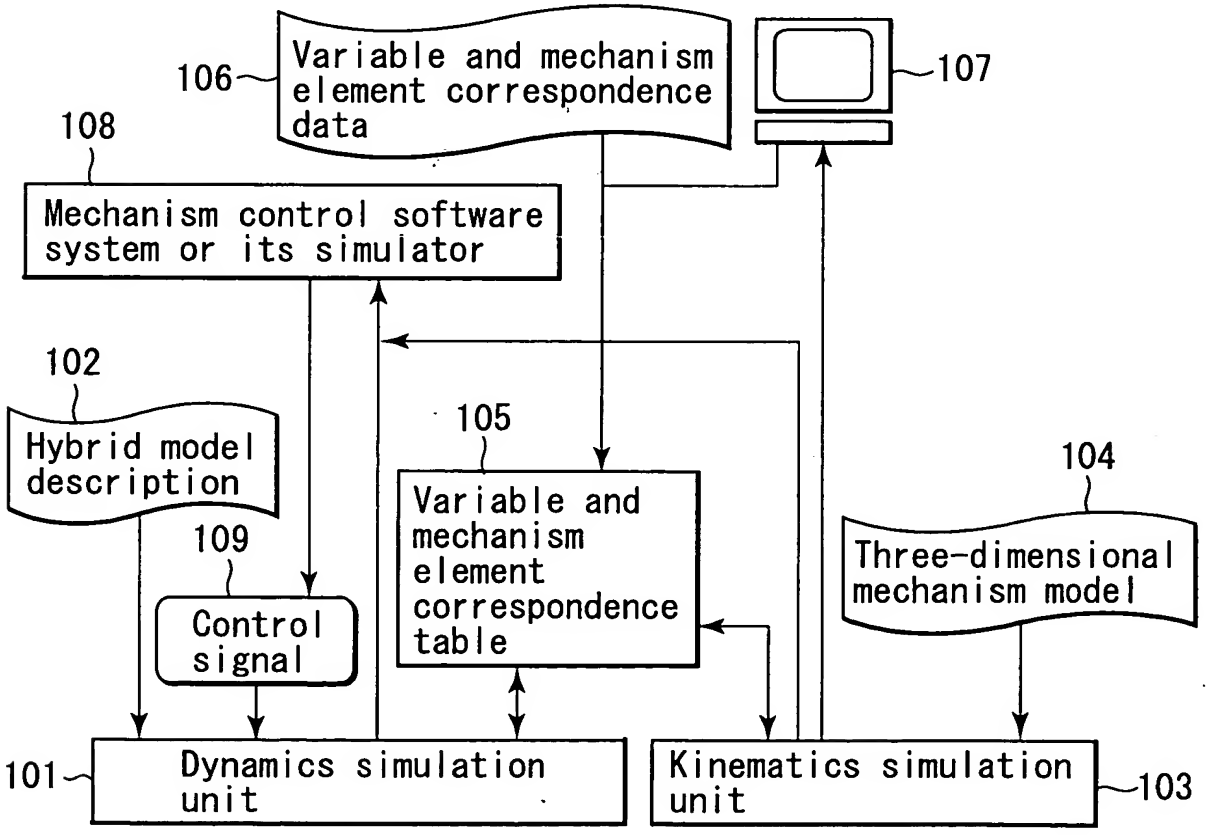


FIG. 2

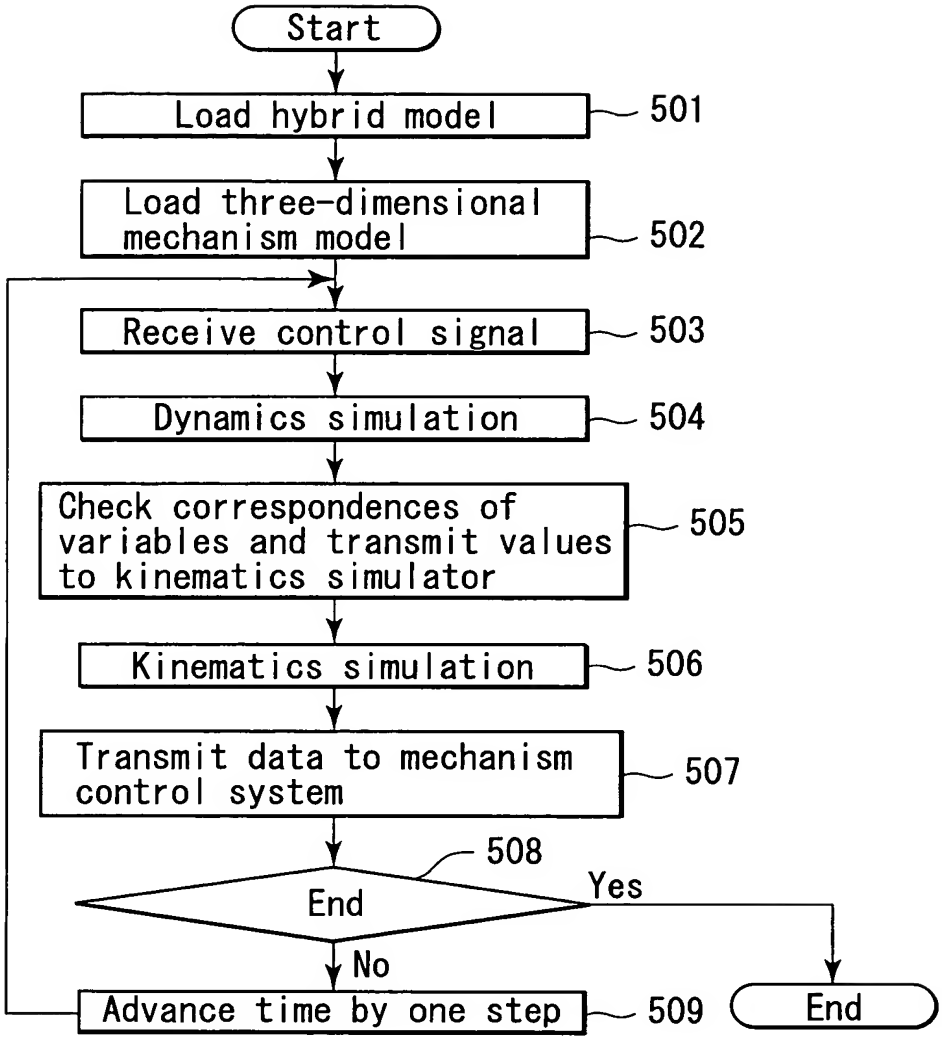

Start
Load hybrid model
501
Load three-dimensional mechanism model
502
Receive control signal
503
Dynamics simulation
504
Check correspondences of variables and transmit values to kinematics simulator
505
Kinematics simulation
506
Transmit data to mechanism control system
507
508
End
Yes
No
Advance time by one step
509
End

FIG. 3

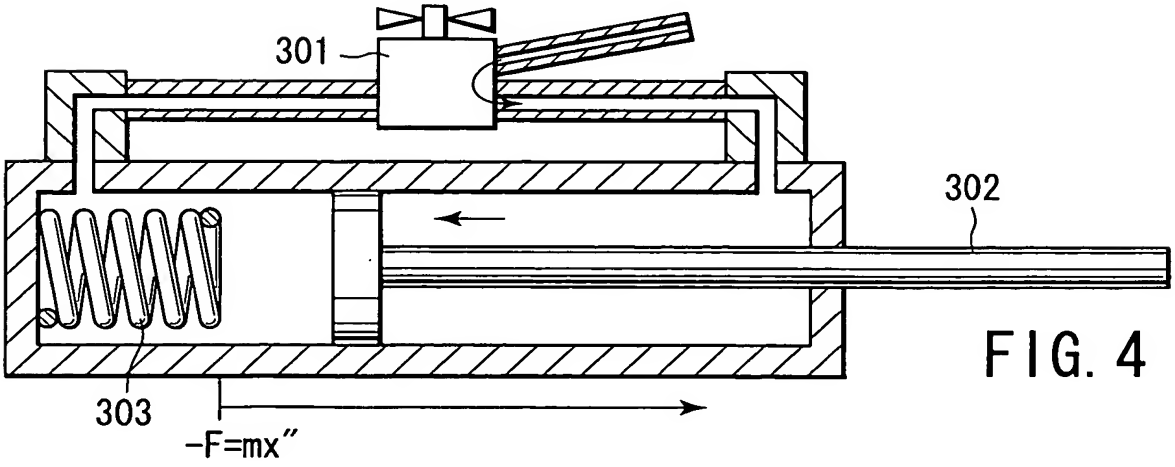

301
302
303
-F=mx″

FIG. 4

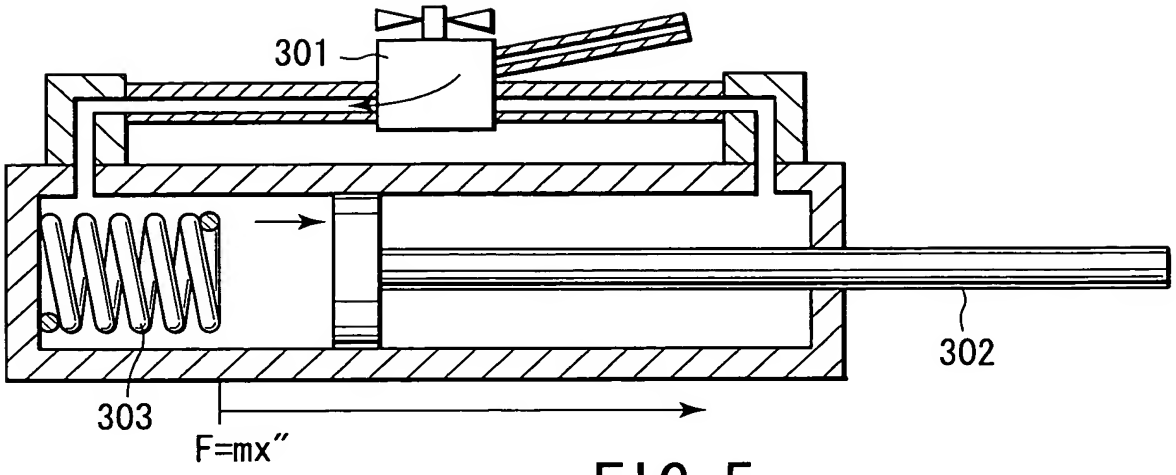


FIG. 5

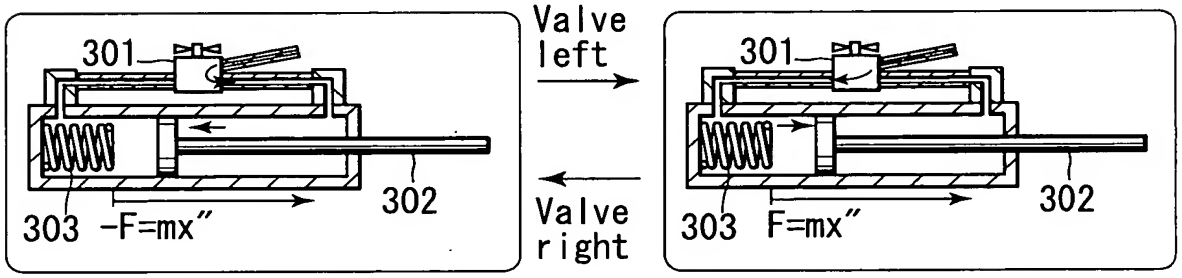


FIG. 6

```
L1…#define m 1
L2…#define f 100
L3…Right ev1
L4…wait 50 do Left ev2
L5…always if Left ev3 then do always F=m*x″ eq1 watching Right,
L6…always if Right ev4 then do always -F=m*x″ eq2 watching Left,
L7…sample(x),
L8…x=0, x′=0,
```

FIG. 7

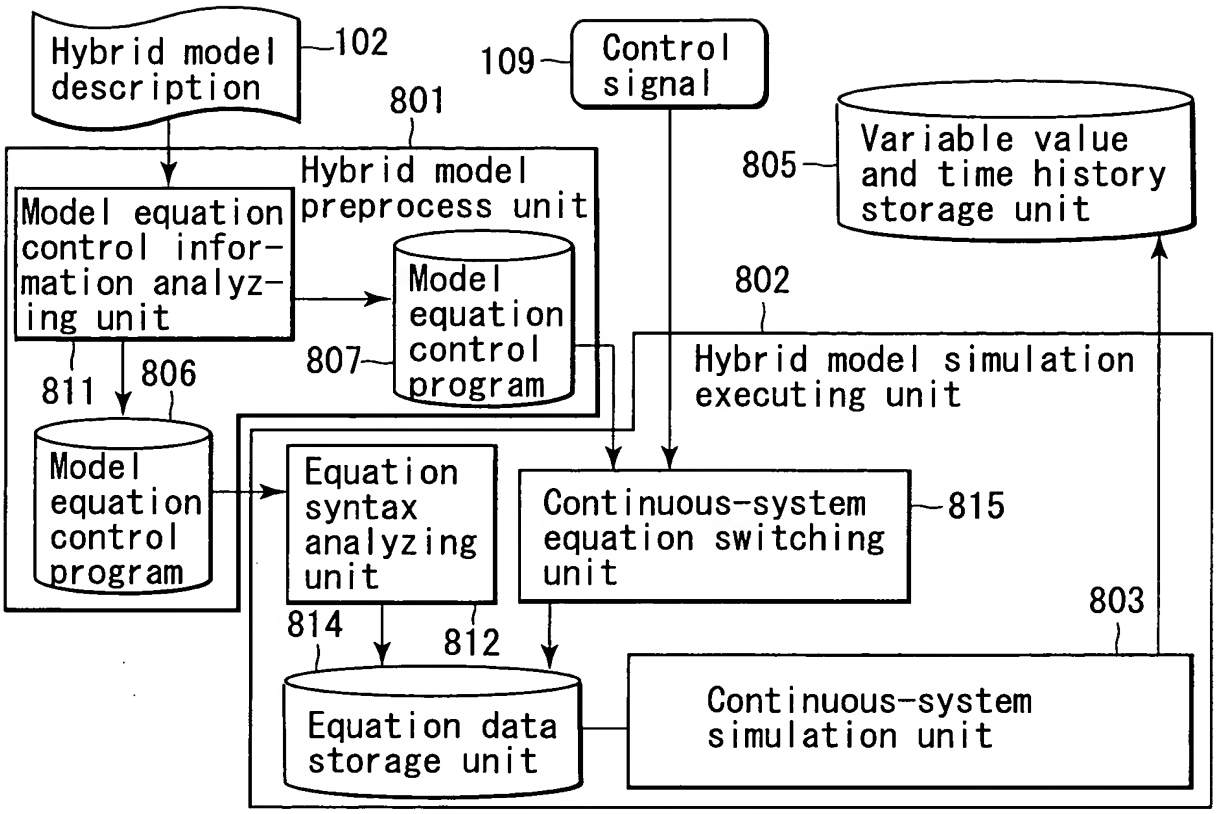

Hybrid model description
102
109
Control signal
801
Hybrid model preprocess unit
805
Variable value and time history storage unit
Model equation control information analyzing unit
Model equation control program
807
802
Hybrid model simulation executing unit
811
806
Model equation control program
Equation syntax analyzing unit
Continuous-system equation switching unit
815
803
814
812
Equation data storage unit
Continuous-system simulation unit

FIG. 8

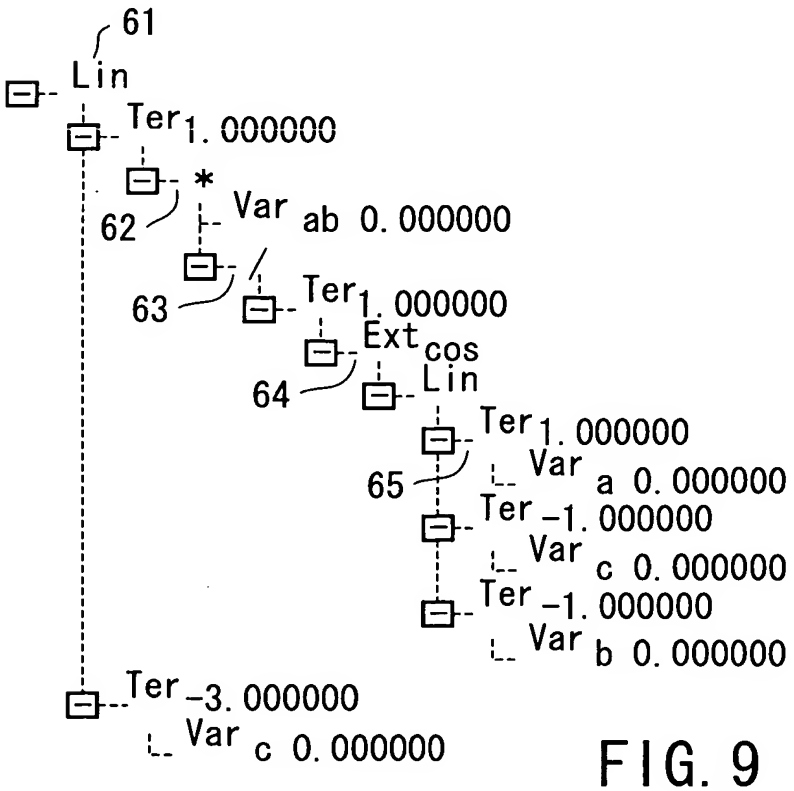

61
Lin
Ter 1.000000
*
62
Var ab 0.000000
63
Ter 1.000000
Ext cos
64
Lin
Ter 1.000000
65
Var a 0.000000
Ter -1.000000
Var c 0.000000
Ter -1.000000
Var b 0.000000
Ter -3.000000
Var c 0.000000

FIG. 9

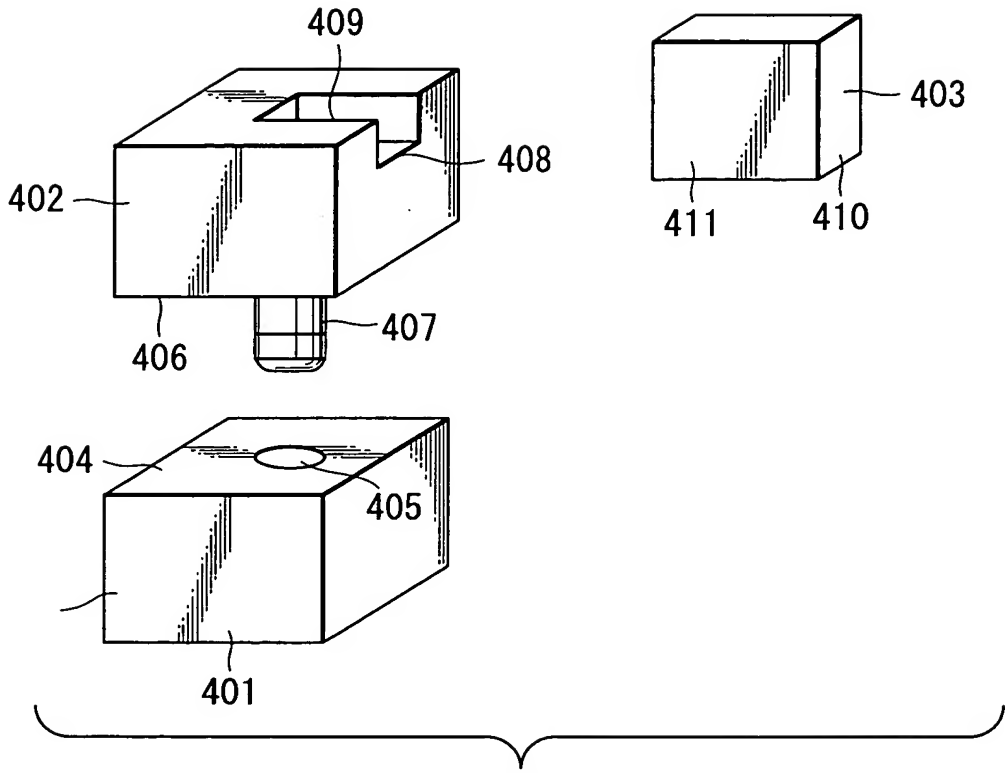

409
408
402
403
411
410
406
407
404
405
401

FIG. 10

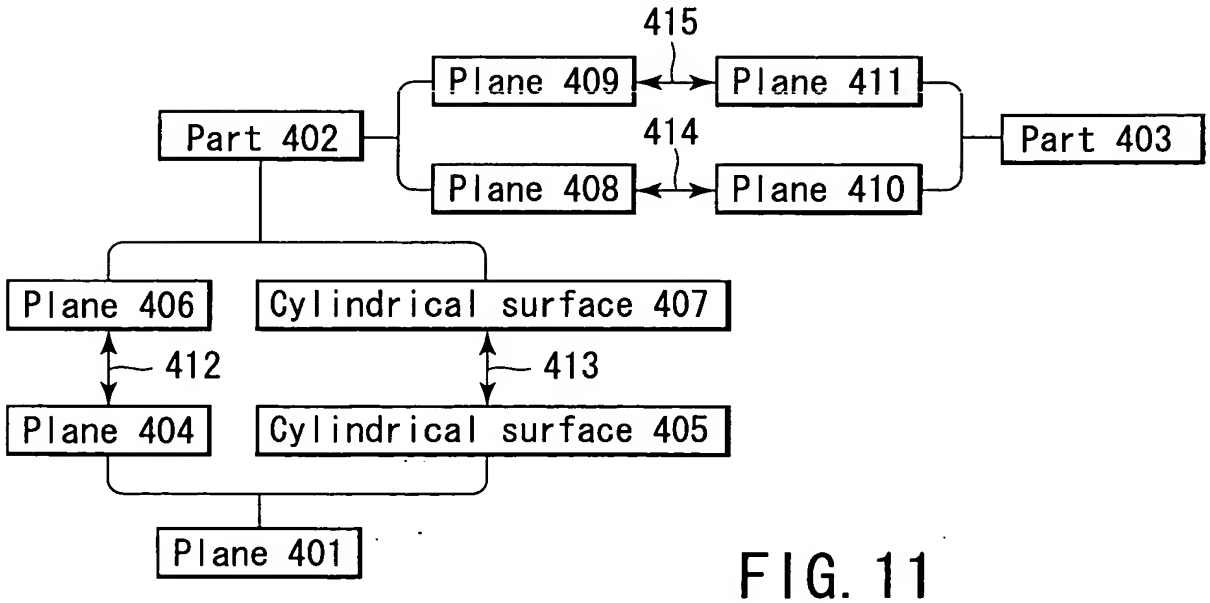

415
Plane 409
Plane 411
Part 402
414
Part 403
Plane 408
Plane 410
Plane 406
Cylindrical surface 407
412
413
Plane 404
Cylindrical surface 405
Plane 401

FIG. 11

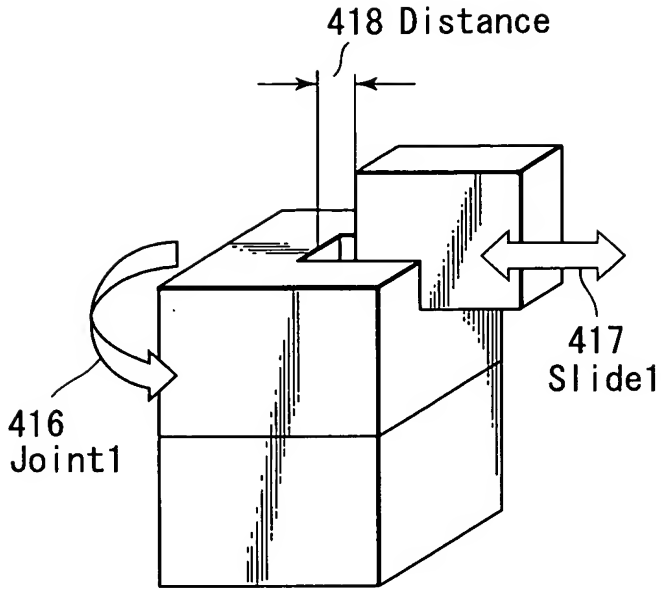


FIG. 12

Variable	Mechanical element name
x	Slide1
y	Joint1

FIG. 13

```
cont a,b;
cont sin(cont x);

a=0;
b=0;
always {
  b' =0.2;
  a' =sin(b);
}
sample(a);
```

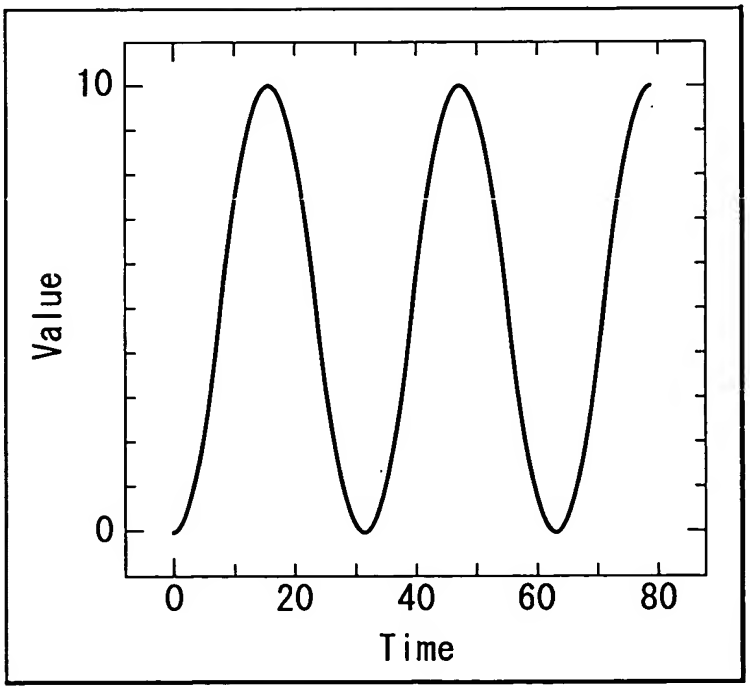
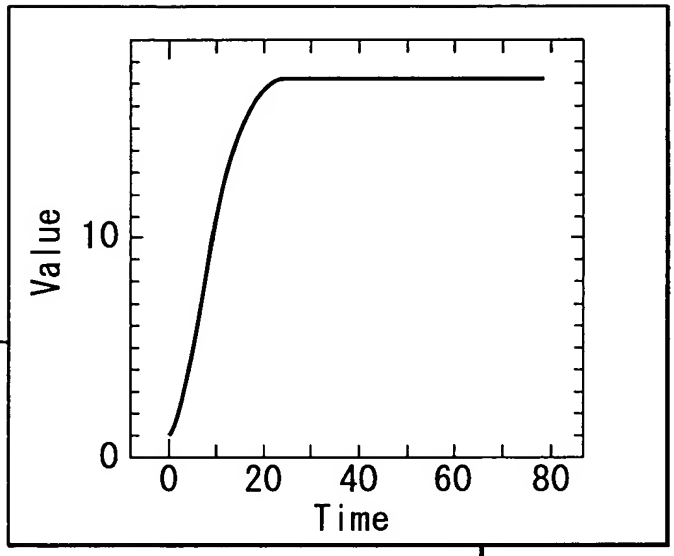


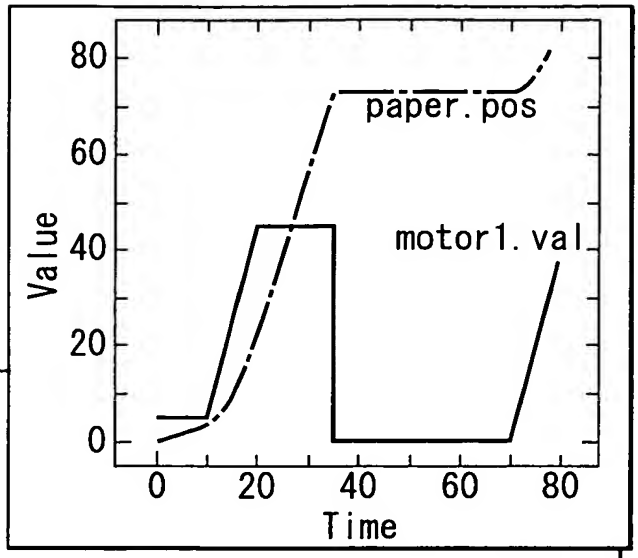
FIG. 14




```
class DCML_Class
{
   public cont m_a;
   DCML_Class(cont a) {
      m_a = a;
      m_a' = 0.01;
   }
   def {
      do {
        always m_a''+(1/4)*m_a'+0.03*m_a = 1/2;
      } watching(m_a' = 0);

      when(m_a' = 0) always {m_a' = 0; }
   }
};
DCML_Class app1(1);
sample(app1.m_a);
```

FIG. 15



S1

```
cont time;
time = 0;
always {time' = 1;}

class Motor
{
   Motor(cont init_val,cont acc_end,cont end) {
     val_     =init_val;
     acc_end_  =acc_end;
     end_      =end;
   }
   public event go_;
   private cont end_,acc_end_;
   public cont val_,start_time_;
   def {
     always {end_' = 0;acc_end_' = 0;start_time_' = 0;}
     do  {always val_' = 0; } watching(go_);
     when(go_)start_time_ = time;
     when(go_)always {
         do {always val_' = 4;} watching (time=start_time_+acc_end_);
         when (time=start_time_+acc_end_)
               do {always val_' = 0;} watching (time=start_time_+end_)
           when (time = start_time_+end_)always val_ = 0;
        }
    }
};

class Paper
{
     public cont val_,pos_;
     def {
          pos_      =0;
          always pos_' = val*0.075;
     }
};
```

```
always paper.val_ = motor1.val_;
when (time = 10)motor1.go_;
when (time = 70)motor1.go_;

Paper paper;
Motor motor1 (5.0,10.,25.);

sample (motor1.val_);
sample (paper.pos_);
```

S2

FIG. 16

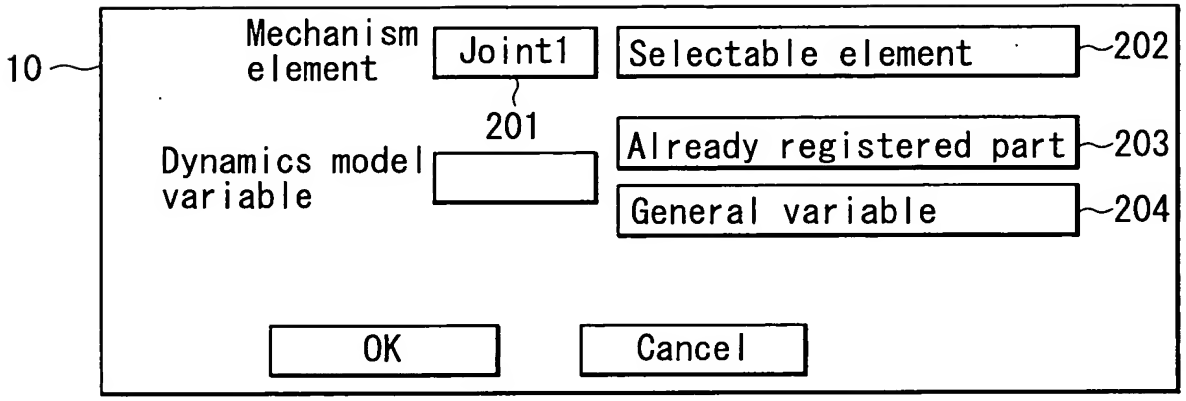


FIG. 17

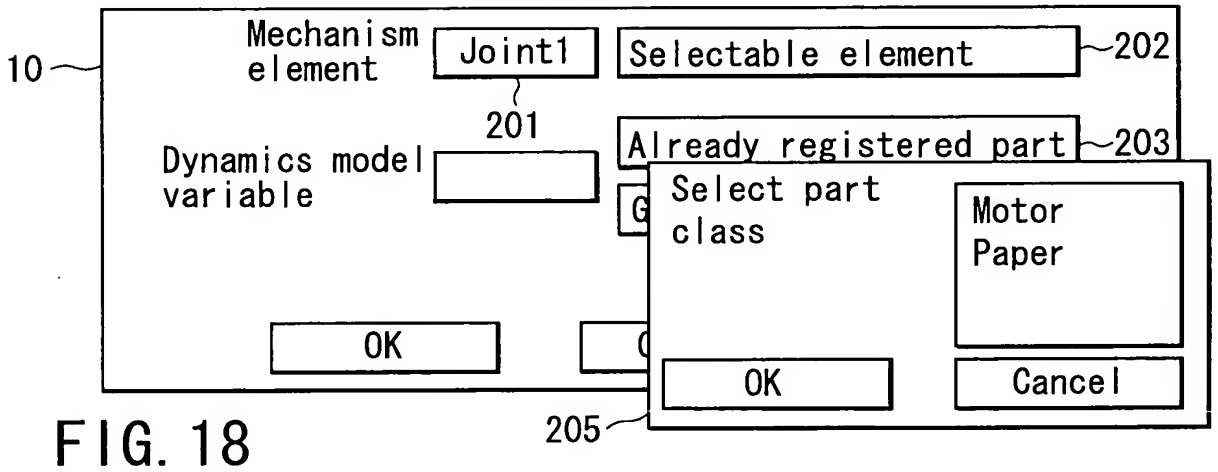


FIG. 18

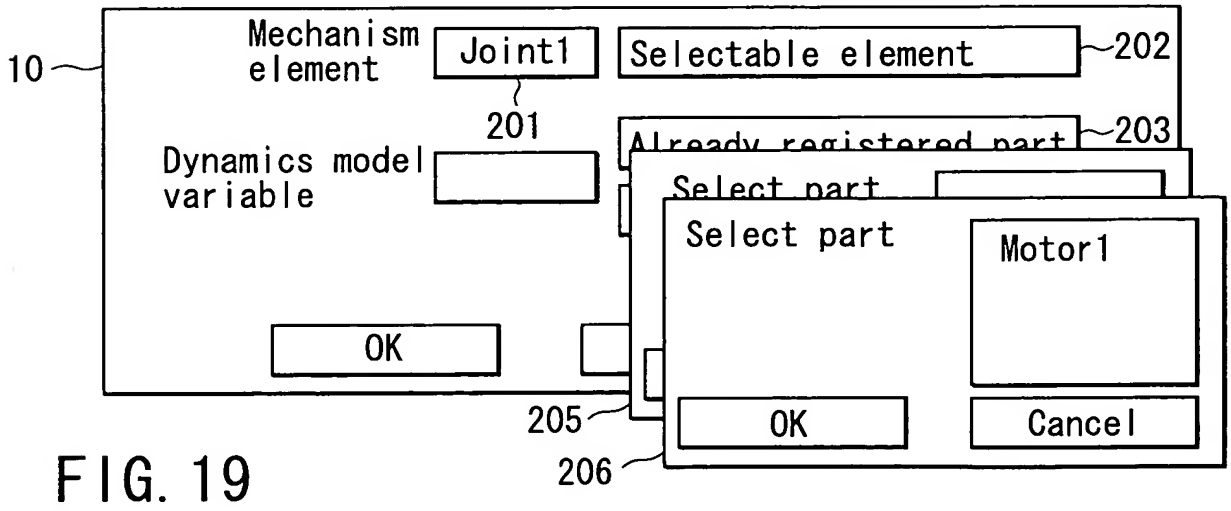


FIG. 19

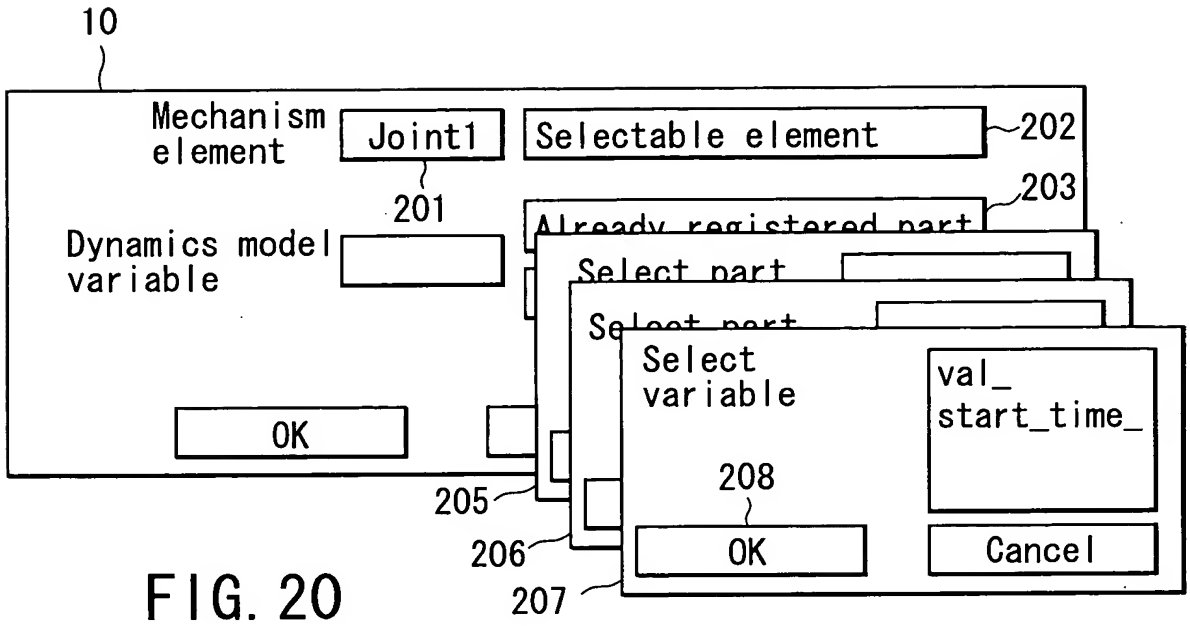

10
Mechanism element
Joint1
Selectable element
202
201
203
Dynamics model variable
Select part
Select variable
val_ start_time_
OK
208
205
206
OK
Cancel
207

FIG. 20

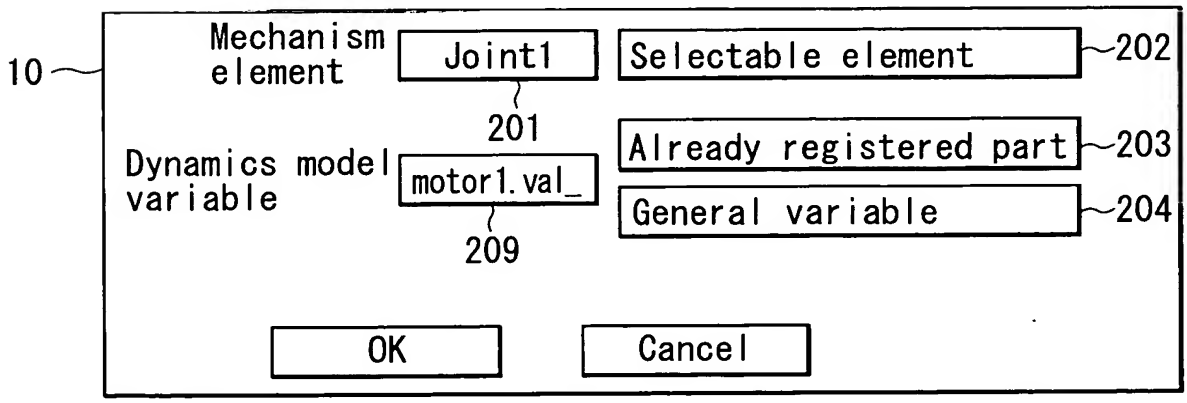

10
Mechanism element
Joint1
Selectable element
202
201
Already registered part
203
Dynamics model variable
motor1.val_
General variable
204
209
OK
Cancel

FIG. 21